# How to create a real-time chat app in Universal Windows Platform using ASP.NET SignalR

## Introduction

This sample demonstrates how to create a real-time chat app in Universal Windows Platform using ASP.NET SignalR.

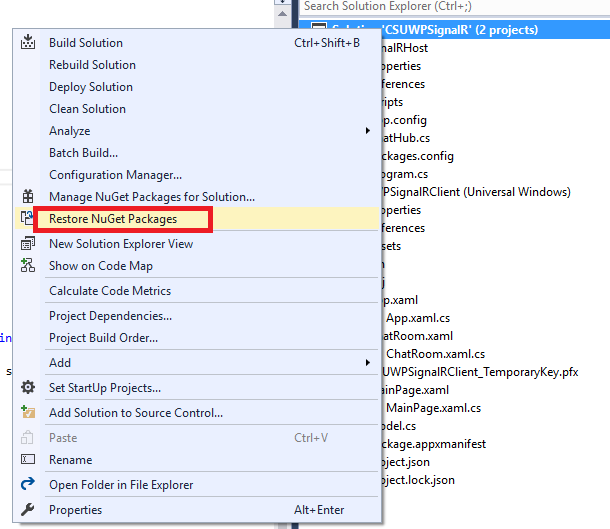
SignalR allows bi-directional communication between server and client. Servers can now push content to connected clients instantly as it becomes available.

In this sample, we create a self-host SignalR console app as the server and a UWP app as the client.

## Building the sample

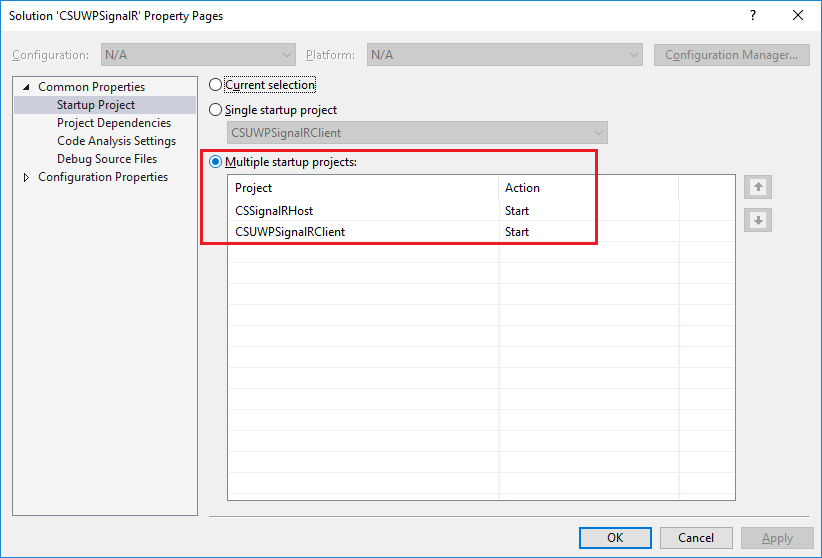
This sample should be run in Microsoft Visual Studio 2015 version and Windows 10.

Before you build the project, make sure you have restored all the packages in the project.



## Running the sample

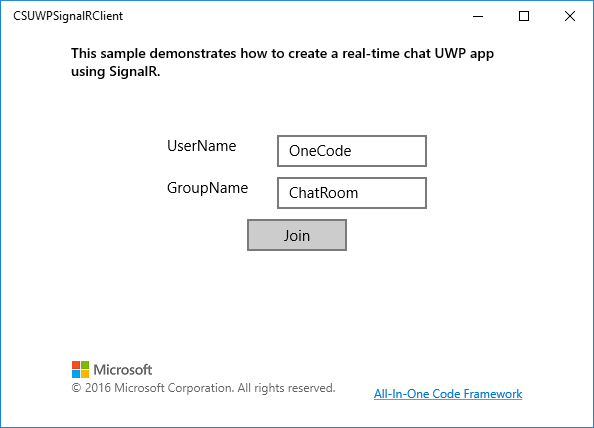
1. Open solution in Visual Studio 2015.
2. Right-click on the solution, and select Set Startup Projects.... Select the Multiple startup projects radio button, and set both projects' Action to Start.



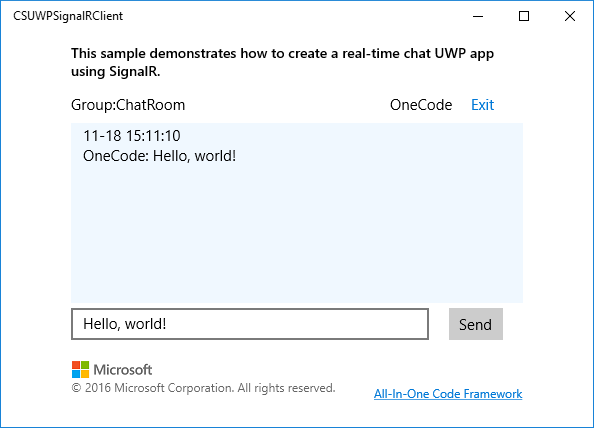
1. Then do one of the following:

* Click the Start Debugging button on the toolbar.
* Click Start Debugging in the Debug menu.
* Press F5.

1. Input user name and group name then click **Join** button.



1. In the chat page, input messages and then click **Send.**



1. Click **Exit** to exit the chat room.

**Using the code**

CSSignalRHost project

[HubName("ChatHub")]

public class ChatHub : Hub

{

/// <summary>

/// add connection to group

/// </summary>

/// <param name="groupName"></param>

/// <returns></returns>

public Task JoinGroup(string groupName)

{

return Groups.Add(Context.ConnectionId, groupName);

}

/// <summary>

/// remove connection from group

/// </summary>

/// <param name="groupName"></param>

/// <returns></returns>

public Task LeaveGroup(string groupName)

{

return Groups.Remove(Context.ConnectionId, groupName);

}

/// <summary>

/// send message to the connections in the group.

/// </summary>

/// <param name="groupName"></param>

/// <param name="userName"></param>

/// <param name="message"></param>

/// <param name="sendTime"></param>

public void SendToGroup(string groupName, string userName, string message, DateTime sendTime)

{

Clients.Group(groupName).ReceiveMessage(userName, message, sendTime);

}

}

static void Main(string[] args)

{

string url = "http://127.0.0.1:8080";

using (WebApp.Start(url))

{

Console.WriteLine($"running... on {url}");

Console.ReadLine();

}

}

public void Configuration(IAppBuilder app)

{

app.UseCors(CorsOptions.AllowAll);

app.MapSignalR();

}

CSUWPSignalRClient project:

Send message handler

string message = this.tbxMessage.Text.Trim();

if (message.Length > 0)

{

myApp.MyHubProxy.Invoke("SendToGroup", groupName, userName, message, DateTime.Now);

}

## More information

[ASP.NET SignalR](https://www.asp.net/signalr/overview/getting-started)